**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软工2303班

学 号： 8209230324

姓 名： 朱正杰

**实验报告成绩：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验** | **实验一** | **实验二** | **实验三** | **实验四** | **实验五** | **总评** |
| **成绩** |  |  |  |  |  |  |

**批阅教师：**

**实验三 函数**

**一、实验目的**

本实验主要培养、训练学生对函数的理解，要求：

1. 掌握函数的定义、声明的方法；

2. 掌握函数的编写要求；

3. 掌握函数的调用方法；

4. 掌握函数参数的传递方法；

5. 掌握变量的作用域；

6. 掌握多文件编程方法。

**二、实验内容与要求**

1、输入自然数m和n，

（1）求他们的最大公约数（或称最大公因数）。

要求输入、输出在主函数中进行，求公约数由函数实现。

1. 在函数中求最大公约数与最小公倍数。（提示：使用引用参数）

2. 编写程序满足：声明一个函数，判断一个整数是否为素数，使用如下函数头：

bool is\_prime(int num) ,如果num是素数函数返回true，否则返回false；

利用函数is\_prime找出前200个素数，并按每行10个输出：

     2     3      5      7    11    13    17    19    23    29

3、编程实现摄氏温度到华氏温度的转换：

编写一个头文件，包含下面两个函数：

double celsius\_to\_fah(double cel)    //摄氏温度到华氏温度

double fahrenheit\_to\_cels(double fah) //华氏温度到摄氏温度

实现头文件，并编写测试程序，调用函数显示如下结果：

Celsius    Fahrenheit   |   Fahrenheit       Celsius

40.0       105.0        |   120.0            48.89

39.0       102.0        |   110.0            43.33

……       ……        |   ……             ……

31.0        87.8        |   30.0             -1.11

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytemperature.h只有函数声明；函数定义写在另一CPP文件mytemperature.cpp）

4、创建名为mytriangle.h的头文件，包括：

bool is\_valid(double side1,double side2,double side3)

double\_area(double side1,double side2, double side3)

面积=sqrt(s(s-side1)(s-side2)(s-side3))

其中s=(side1+side2+side3)/2

写测试程序：读取三角形三边长，如输入合法，计算面积，否则输出错误信息。

（测试程序为主模块，即main( )函数所在的CPP文件，头文件mytriangle.h只有函数声明；函数定义写在另一CPP文件mytriangle.cpp）

**3与4选一个完成**

5、猴子吃桃：猴子第一天摘若干桃子，当即吃了一半，还不过瘾，又吃了一个。第二天又将剩下的桃子吃掉一半，又多吃一个，以后每天如此，到第10天，发现只剩最后一个桃子，问，第一天猴子共摘多少桃子（用递归实现）。

**三、实验思考题**

1. 本实验中函数中返回的值为什么与函数类型一致？

**函数本没有类型，只是为了识别返回值的类型，才在声明中在函数前面也加了这个类型，一遍编译程序把返回值以这个指定类型来处理。所以，函数中返回值的类型只能与函数声明时的类型一致。**

2. 本实验中主函数调用函数时采用的是何种传递方式？

**传用参数的传递方式。**

**四、算法分析，程序结果**

**3.1：(1)**

#include<iostream>

using namespace std;

void max1(int a, int b) {

int c = 1;

int temp = 0;

while (c <= a && c <= b) {

if (a % c == 0 && b % c == 0) {

temp = c;

}

c++;

}

}

int main() {

unsigned int a, b;

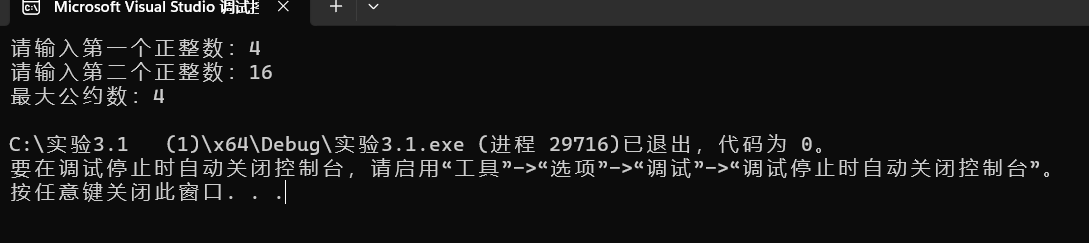
cout << "请输入第一个正整数：";

cin >> a;

cout << "请输入第二个正整数：";

cin >> b;

max1(a, b);

 cout << "最大公约数：" << temp << endl;

}

(2)

#include<iostream>

using namespace std;

int& max1(int a, int b) {

int c = 1;

static int temp1 = 0;

while (c <= a && c <= b) {

if (a % c == 0 && b % c == 0) {

temp1 = c;

}

c++;

}

return temp1;

}

int& min1(int a, int b) {

int d = a\*b;

static int temp2 = 0;

while (d >= a && d >= b)

{

if (d % a == 0 && d % b == 0) {

temp2 = d;

}

d--;

}

return temp2;

}

int main() {

unsigned int a, b;

cout << "请输入第一个正整数：";

cin >> a;

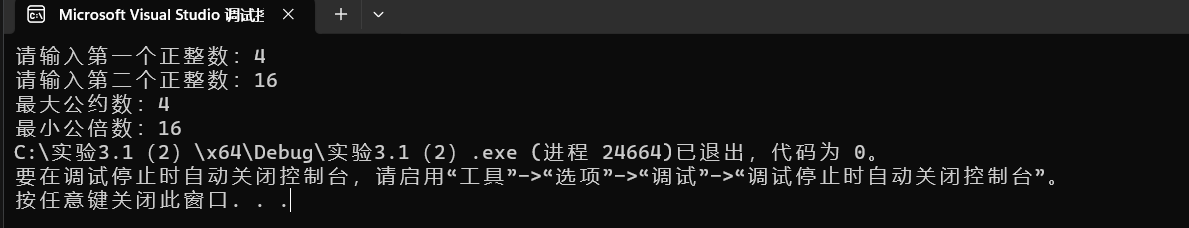
cout << "请输入第二个正整数：";

cin >> b;

cout << "最大公约数：" << max1(a, b) << endl;

cout << "最小公倍数：" << min1(a,b);

}



**3.2:**

#include<iostream>

using namespace std;

bool is\_prime(int n);

int main() {

int n=1;

int z = 0;

while (1) {

n++;

if (is\_prime(n))

{

z++;

cout << n<<'\t';

if(z%10==0)

{

cout << endl;

}

}

if(z%200==0)

{

break;

}

}

}

bool is\_prime(int n)

{

for (int i = 2; i <= n - 1; i++)

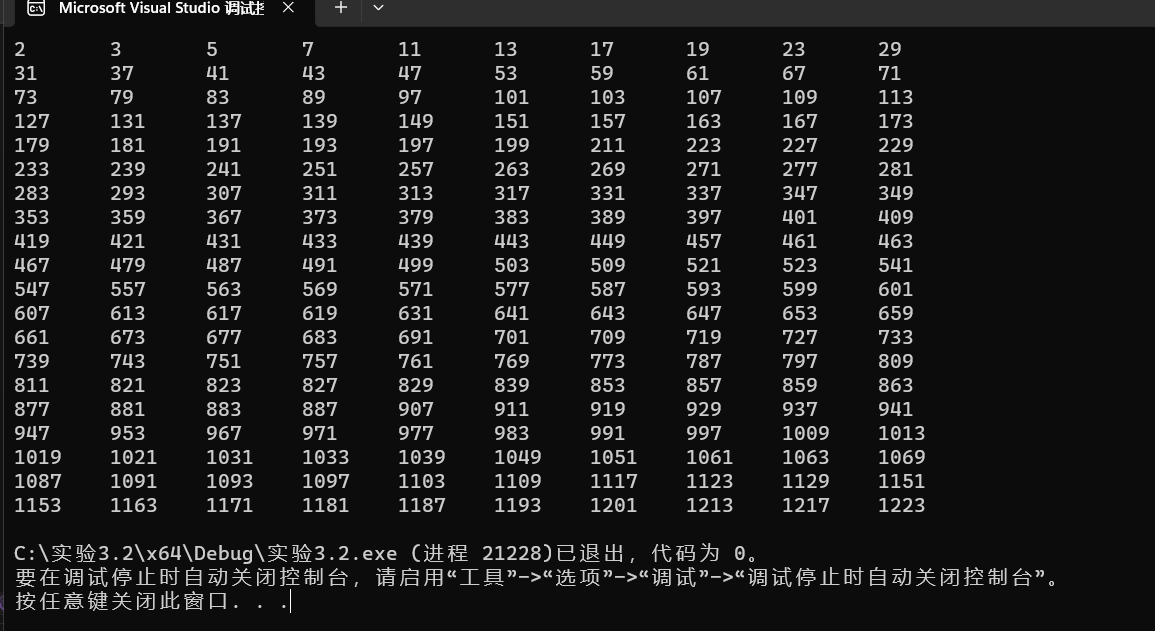
{

if (n % i == 0) {

return false;

}

}

 return true;

}

**3.4:**

#pragma once

#include<iostream>

using namespace std;

bool is\_valid(double a, double b, double c);

double are\_a(double a, double b, double c);

#include"mytriangle.h"

int main() {

double a, b, c;

cout << "请输入第一条边的长：";

cin >> a;

cout << "请输入第二条边的长：";

cin >> b;

cout << "请输入第三条边的长：";

cin >> c;

is\_valid(a, b, c);

}

#include"mytriangle.h"

bool is\_valid(double a, double b, double c) {

if (a + b > c && a + c > b && b + c > a)

{

cout << "三角形的面积是：" << are\_a(a, b, c);

return true;

}

else cout << "你输入的边无法构成三角形。";

return false;

}

double are\_a(double a, double b, double c) {

int d = (a + b + c) / 2;

return sqrt(d \* (d - a) \* (d - b) \* (d - c));

}

**3.5:**

#include<iostream>

using namespace std;

int day(int a,int i)

{

int temp;

temp = (a + 1) \* 2;

a = temp;

i++;

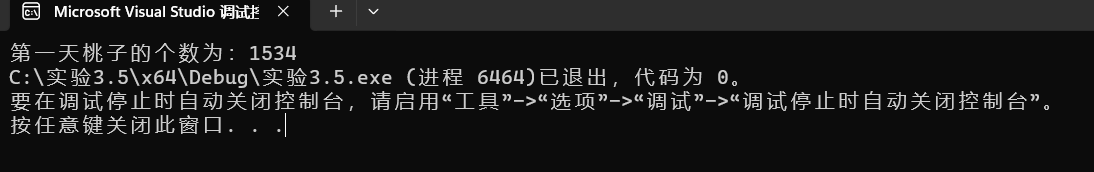
if (i < 9) { day(a, i); }

else return a;

}

int main() {

cout<<"第一天桃子的个数为：" <<day(1 ,0);

}

**五、遇到的问题与解决方法**

**问题：（1）头文件的运用并不熟练**

**（2）3.5不清楚第十天到底是吃前还是吃后**

**解决方法：（1）上网查看教程。**

**（2）询问同学，并在网上搜查原题，得以解决问题。**

**六、体会**

**（1）编程语言的学习前后关联是非常强的，我个人觉得实验1和2的**

**学习对实验3的进行有很大的帮助**

**（2）一定要先看清要求再开始行动,就算写的再好，与要求不符的代码也只能得0分。**

**（3）要多去发掘已有知识的特殊用法，会让解决问题变得事半功倍。**

**实验四 数组与指针**

【**实验目的**】

1、进一步加深对数组的理解，掌握数组的定义方法；

2、掌握数组的处理方法、数组作为函数参数的使用方法，以及搜索与排序的应用。

3、掌握指针的概念、指针变量定义格式以及指针的运算；

4、掌握指针与数组、函数的关系；

5、理解内存动态分配的含义、熟练掌握内存动态分配方法；

6、掌握递归函数的定义方法。

【实验内容与步骤】

**（一）数组**

1、打印不同的数：

编写一个程序，读入10个数，输出其中不同的数（即如果一个数出现多次，只打印一次）。

提示：读入的数如果是一个新的值，则将其存入一个数组。否则，将其丢弃。输入完毕后，数组中保存的就是不同的数。

下面是一个运行样例：

Enter

Enter ten numbers: 1 2 3 2 1 6 3 4 5 2

The distinct numbers are: 1 2 3 6 4 5

2、起泡排序：

利用起泡排序算法编写一个排序函数。起泡排序算法分若干趟对数组进行处理。每趟处理中，对相邻元素进行比较。若为降序，则交换；否则，保持原顺序。此技术被称为起泡排序（bubble sort）或下沉排序（sinking sort），因为较小的值逐渐地“冒泡”到上部，而较大值逐渐下沉到底部。

算法可描述如下：

bool changed = true;

do

{

changed = false;

for (int j = 0; j < listSize – 1; j++)

if (list[j] > list[j+1])

{

swap list[j] with list[j+1];

changed = true;

}

} while (changed);

很明显，循环结束后，列表变为升序。容易证明do循环最多执行listSize – 1次。

编写测试程序，读入一个含有10个双精度数字的数组，调用函数并显示排列后的数字。

3、游戏：存物柜问题：

一个学校有100个存物柜，100个学生。开学第一天所有存物柜都是关闭的。第一个学生（记为S1）来到学校后，打开所有的存物柜。第二个学生S2，从第二个存物柜（记为L2）开始，每隔两个存物柜，将它们关闭。第三个学生S3从第三个存物柜L3开始，每隔三个，将它们的状态改变（开着的关上，关着的打开）。学生S4，从L4开始，每隔四个改变它们的状态。学生S5，从L5开始，每隔五个改变状态。依此类推，直至学生S100改变L100的状态。

当所有学生完成这个过程，那些存物柜是开着的？编写一个程序求解此问题，显示所有开着的柜子号码，号码之间用一个空格隔开。

提示：使用一个100个布尔型元素的数组，每个元素代表存物柜是开（true）或关（false）。最初所有的储物柜都是关闭的。

4、合并两个排列好的数组：

编写如下函数，合并两个排列好的数组，形成一个新的排列好的数组。

void merge(const int list1[], int size1, const int list2[], int size2, int list3[])

使用size1+size2次比较实现函数。编写测试程序，提示用户输入两个排列好的数组，并显示合并以后的数组。下面是一个运行样例。注意，输入数据的第一个数字是数组的元素数，而不是数组的一部分。假定数组大小不超过80。

Enter

Enter

Enter list1: 5 1 5 16 61 111

Enter list1: 4 2 4 5 6

The merged list is 1 2 4 5 5 6 16 61 111

5、检验子串：

编写如下函数，检验C字符串s1是否是C字符串s2的子串。如果匹配，返回s1在s2中的下标，否则返回–1。

int indexOf(const char s1[], const char s2[])

编写测试程序，读入两个C字符串，检验C字符串s1是否是C字符串s2的子串。下面是程序的运行样例：

Enter

Enter

Enter

Enter the first string: welcome

Enter the second string: We welcome you!

indexOf(“welcome”, “We welcome you!”) is 3

Enter

Enter the first string: welcome

Enter the second string: We invite you!

indexOf(“welcome”, “We invite you!”) is –1

6、字符串中每个字母出现的次数：

请使用如下函数头编写函数，数出字符串中每个字母出现的次数。

void count(const char s[], int counts[])

counts是一个有26个元素的整数数组。const[0]，const[1]，…，const[25]分别记录a，b，…，z出现的次数。字母不分大小写，例如字母A和字母a都被看作a。

编写测试程序，读入字符串并调用count函数，显示非零的次数。下面是程序的一个运行样例：

Enter

Enter a string: Welcome to New York!

c: 1 times

e: 3 times

k: 1 times

l: 1 times

m: 1 times

n: 1 times

o: 3 times

r: 1 times

t: 1 times

w: 2 times

y: 1 times

**（二）指针**

1、上机验证下列程序的运行结果（有错误的话自己补充完善）

(1) void main()

｛

int i,j,\*pi,\*pj; //此处的\*表示定义指针变量，而非间接运算符

pi=&i;

pj=&j;

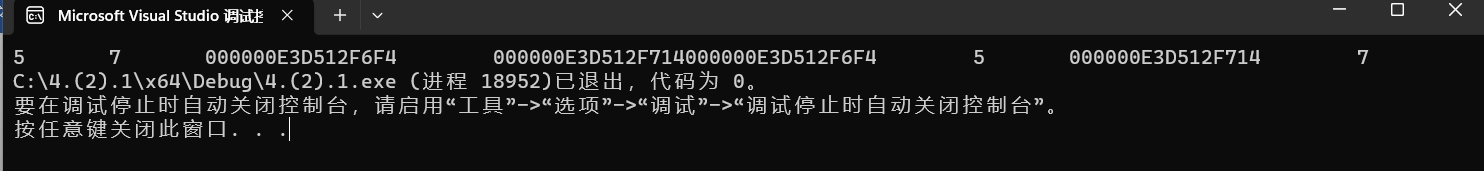
i=5;j=7;

cout<<i<<’\t’<<j<<’\t’<<pi<<’\t’<<pj;

cout<<&i<<’\t’<<\*&i<<’\t’<<&j<<’\t’<<\*&j;

}

运行结果：



上述结果中，pi与&i,pj与&j是地址值，随编译程序而变化，不确定。

(2) int main() //C语言程序，要了解

{

int a[]={1,2,3};

int \*p,i;

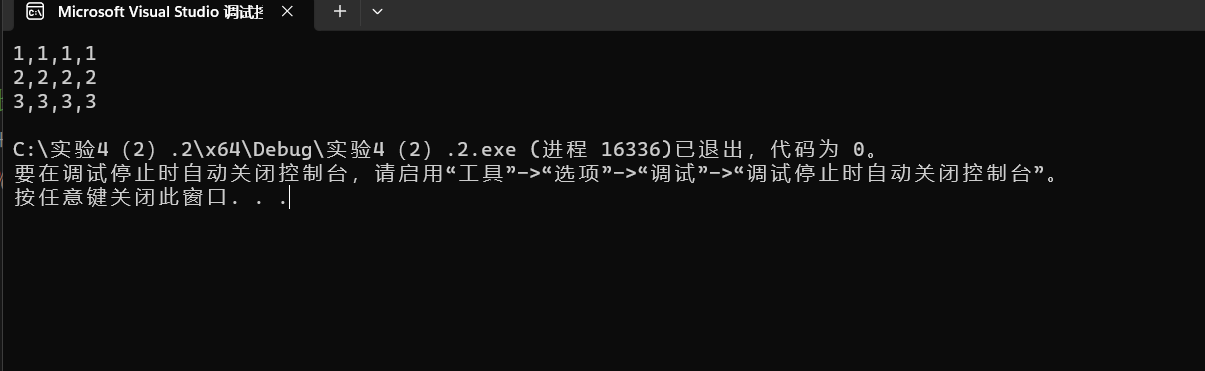
p=a; //将数组a首地址送给p

for (i=0;i<3;i++)

printf("%d,%d,%d,%d\n",a[i],p[i],\*(p+i),\*(a+i)); //与cout功能差不多

}

运行结果：



1,1,1,1

2,2,2,2

3,3,3,3

通过这两道题目，希望学生掌握数组元素与指向数组的指针的不同。

a[i]表示数组中下标为i的元素。

a[i]←p[i]←\*(p+i)←\*(a+i)

a是数组名，表示数组首地址，(p+i)表示数组中第i个元素的地址，\*(p+i) 相当于a[i]。

(3)通过如下的问题理解递归函数的定义与调用（递归未讲，可以后做）

//#include “stdio.h”

void f(char \*st,int i)

{

st[i]=’\0’;

cout<<st; // printf(“%s\n”,st);

if (i>1) f(st,i-1);

}

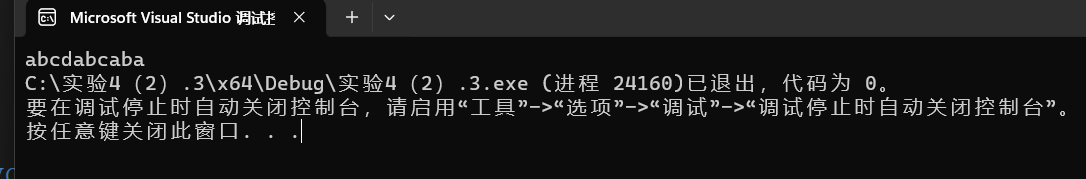
void main()

{

char st[]=”abcd”;

f(st,4);

}

补充完整，运行时输出为\_\_\_\_\_\_\_\_

(4)下面程序的主函数中能保证p[0]输出1，p[1]输出2吗？如何修改以保证之（提示：在函数f中使用new生成动态数组；在main中用delete释放。）

#include<iostream>

using namespace std;

int \*f()

{

int list[]={1,2,3,4};

return list;

}

void main()

{

int \*p=f();

cout<<p[0]<<endl;

cout<<p[1]<<endl;

}

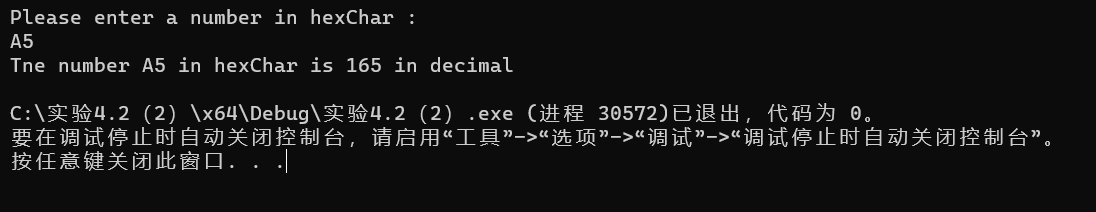
2、程序设计

(1)编写函数检查字符串s1是否为字符串s2的子串，若是，返回第一次匹配的下标，否则返回-1。在主程序中输入字符串s1与s2，调用函数实现。

函数原型：int indexof(const char \*s1,const char \*s2);

(2)编写一个函数将以字符串形式表示的一个16进制数转换为10进制数，并在主函数中测试。函数原型 int parseHex(const char \*const hexString);

如：调用函数 parseHex(“A5”);返回165



1. 主程序中建立一动态数组（使用new），数组元素及元素个数由键盘输入，动态调试观察指针及指针指向的内容；设计一个函数对数组由小到大排序；主程序中用指针方式输出数组元素；最后释放数组内存（delete）。

【完成实验报告】

**实验报告只要求写程序设计部分**

**三、算法分析，程序结果**

**4.（一）.1：**#include<iostream>

using namespace std;

int main()

{

const int size = 10;

int list[size];

cout << "Please enter ten numbers :";

for (int i = 0; i < size; i++)

cin >> list[i];

cout << "The distinct numbers are:";

cout << list[0] << " ";

for (int m = 1; m < size; m++)

{

for (int j = 0; j < m; j++)

{

if (list[m] != list[j])

{

if (j == m - 1)

cout << list[m] << " ";

else

continue;

}

else

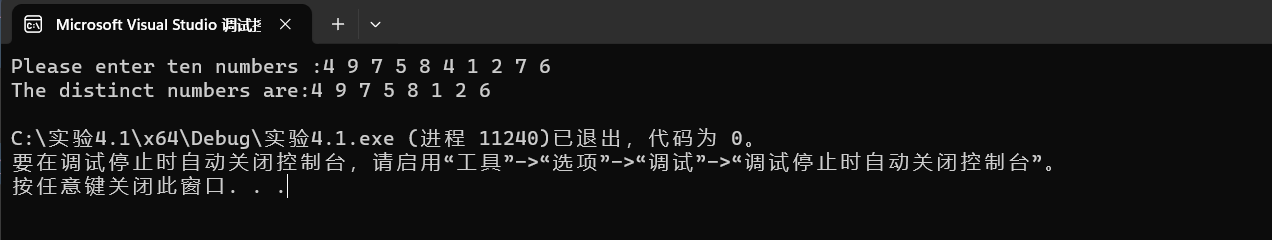
break;

}

}

cout << endl;

return 0;

****}

**4（一）.2**

#include<iostream>

using namespace std;

void sort(double arr[]) {

bool changed = true;

do

{

changed = false;

for (int j = 0; j < 9; j++)

if (arr[j] > arr[j + 1])

{

double temp;

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

changed = true;

}

} while (changed);

}

int main() {

double arr[10];

cout << "请输入数组：";

for (int p = 0; p <= 9; p++)

{

cin >>arr[p];

}

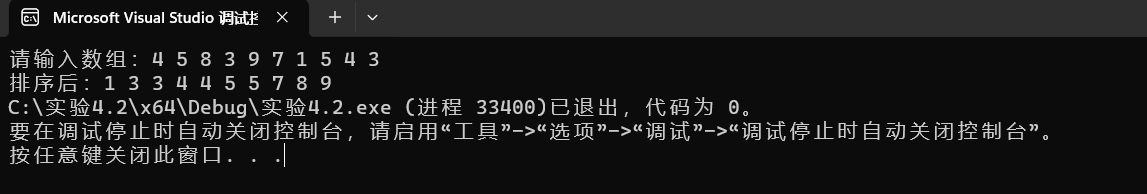
sort(arr);

for (int i = 0; i <= 9; i++)

{

cout << arr[i]<<" ";

}

****}

**4（一）.3**

#include<iostream>

using namespace std;

int main() {

bool arr[100];

for (int i = 0; i < 100; i++) {

arr[i] = false;

}

for (int j = 0; j < 100; j++)

{

for (int p = j; p < 100; p += j + 1)

{

if (arr[p]) {

arr[p] = false;

}

else arr[p] = true;

}

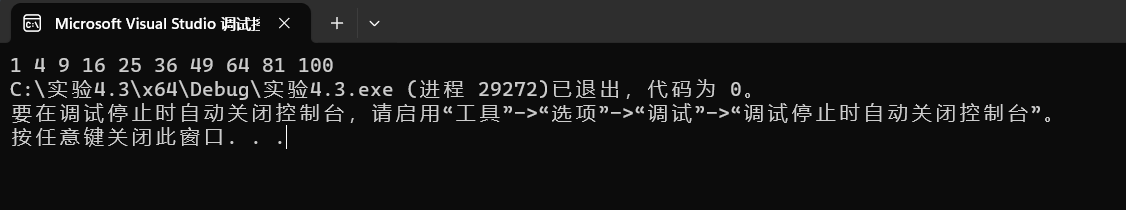
}

for (int n = 0; n < 100; n++) {

if (arr[n])

cout << n + 1<<" ";

}

}

**4（一）.4：**

#include<iostream>

using namespace std;

void merge( int arr1[], int size1, int arr2[], int size2, int arr3[])

{

for (int n = 0; n < size1; n++) {

arr3[n] = arr1[n];

}

int r = 0;

for (int g = size1; g < size1 + size2; g++) {

arr3[g] = arr2[r];

r++;

}

}

void arrange(int arr3[] ,int size1,int size2) {

for (int i = 0; i < size1 + size2 - 1; i++) { //控制n-1趟冒泡

for (int j = 0; j < size1 + size2 - 1 - i; j++)

{

if (arr3[j] > arr3[j + 1]) { //比较相邻的两个元素

int tmp; //临时变量

tmp = arr3[j]; //交换

arr3[j] = arr3[j + 1];

arr3[j + 1] = tmp;

}

}

}

}

int main(){

int size1, size2, size3;

int arr1[40],arr2[40];

cout << "请输入1数组的长度与数字：";

cin >> size1;

for (int i = 0; i <size1 ; i++) {

cin >> arr1[i];

}

cout << "请输入2数组的长度与数字：";

cin >> size2;

for (int j = 0; j < size2; j++) {

cin >> arr2[j];

}

size3 = size1 + size2;

int arr3[80];

merge(arr1, size1, arr2, size2, arr3);

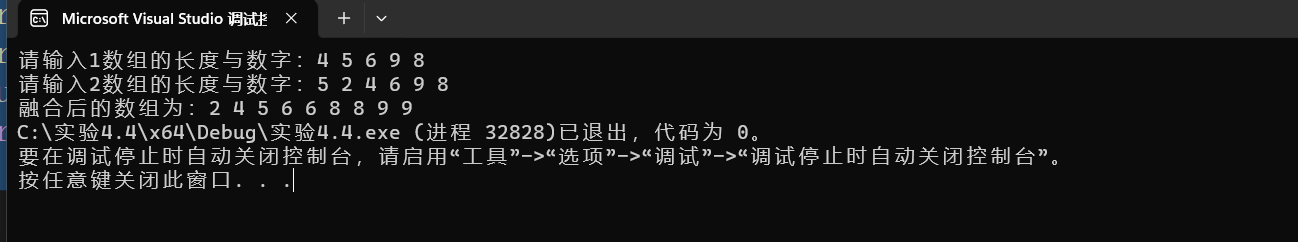
arrange(arr3 ,size1,size2);

cout << "融合后的数组为：";

for (int t = 0; t < size3; t++) {

cout << arr3[t]<<" ";

}

 }

**4（一）.5：**

#include<iostream>

using namespace std;

int indexOf1(char s1[], char s2[]) {

if (s2[0] == 0) {

return 0;

}

else {

for (int i = 0; s1[i]; i++) {

int k = i, j = 0;

for (; s2[j];++k,++j) {

if (s1[k] != s2[j]) {

break;

}

}

if (s2[j] == 0) {

return i;

}

}

return -1;

}

}

int main() {

int size1, size2;

char s1[100];

char s2[100];

cout << "请输入字符串1的长度与字符：";

cin >> size1;

for (int i = 0; i < size1; i++) {

cin >> s1[i];

}

cout << "请输入字符串2：";

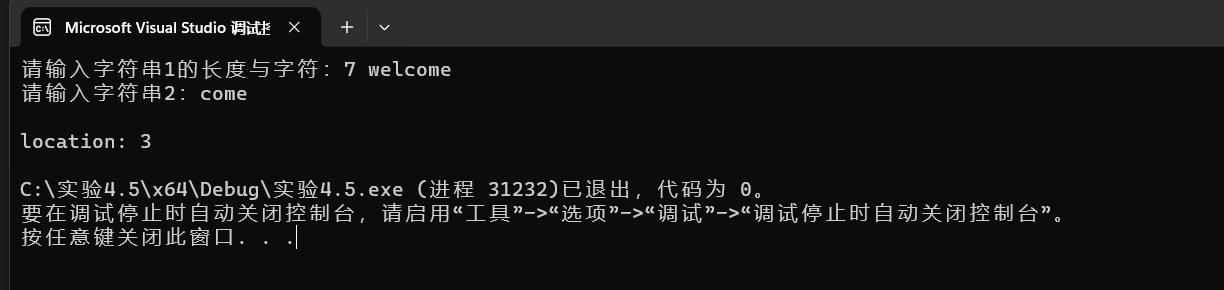
cin >> s2;

cout << endl;

int n = indexOf1(s1,s2);

cout << "location: " << n << endl;

return 0;

}

**4（一）.6：**

#include<iostream>

using namespace std;

void count(const char a[], int counts[])

{

for (int j = 0; j < 100; j++)

{

if ((int)a[j] < 90) {

counts[(int)a[j] - 65]++;

}

else {

counts[(int)a[j] - 97]++;

}

}

}

int main() {

char a[100];

int size1;

cout << "请输入字符串的长度与字符：";

cin >> size1;

for (int i = 0; i < size1; i++) {

cin >> a[i];

}

static int counts[26];

count(a, counts);

for (int k = 0; k < 26; k++)

{

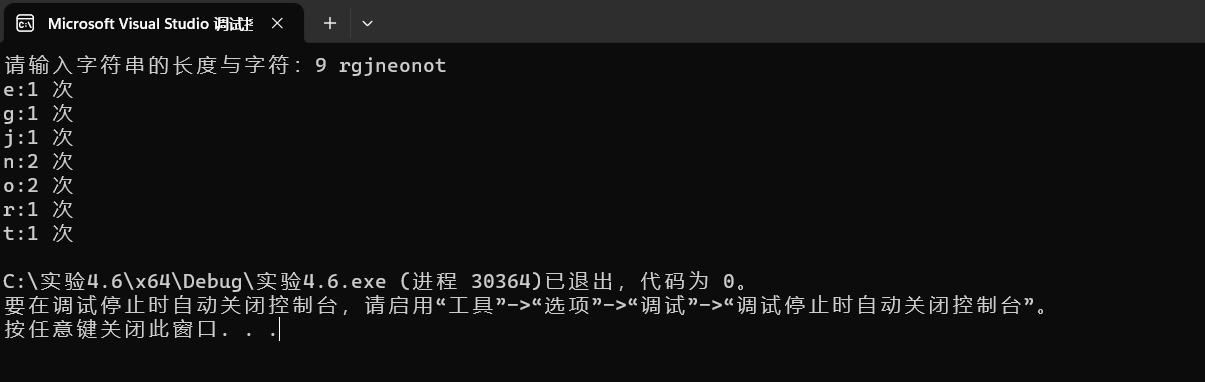
if (counts[k] != 0) {

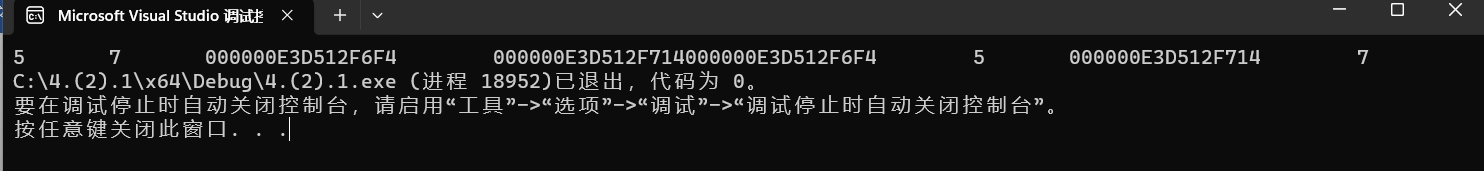
cout << (char)(k + 97) << ':' << counts[k] <<" " << "次" << endl;

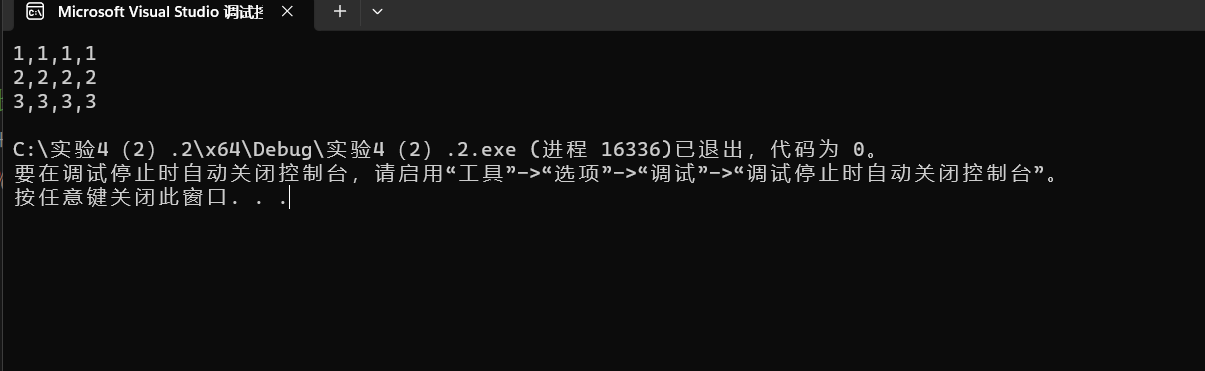
}

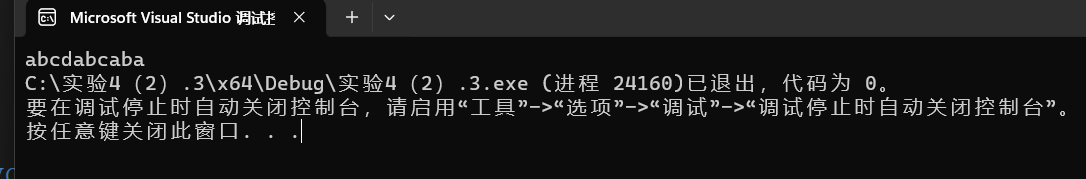
}

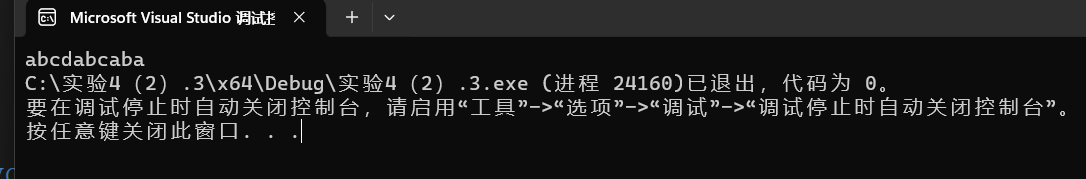
}



**4（二）.1.（1）：**

**4（二）.1.（2）：**

**4（二）.1.（3）：**

**4（二）.1.（4）：**

#include<iostream>

using namespace std;

int\* f()

{

int\* list = new int[4];

list[0] = 1;

list[1] = 2;

list[2] = 3;

list[3] = 4;

return list;

}

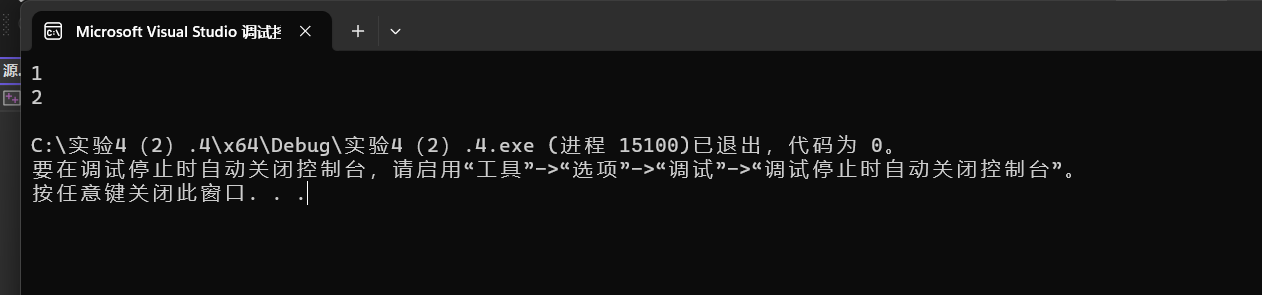
void main()

{

int\* p = f();

cout << p[0] << endl;

cout << p[1] << endl;

**** delete p;

}

**4（二）.2.（1）：**

#include<iostream>

using namespace std;

int indexOf(char \*s1, char\* s2) {

if (s2[0] == 0) {

return 0;

}

else {

for (int i = 0; s1[i]; i++) {

int k = i, j = 0;

for (; s2[j]; ++k, ++j) {

if (s1[k] != s2[j]) {

break;

}

}

if (s2[j] == 0) {

return i;

}

}

return -1;

}

}

int main() {

char s1[40];

char s2[40];

cout << "s1= ";

cin >> s1;

cout << endl;

cout << "s2= ";

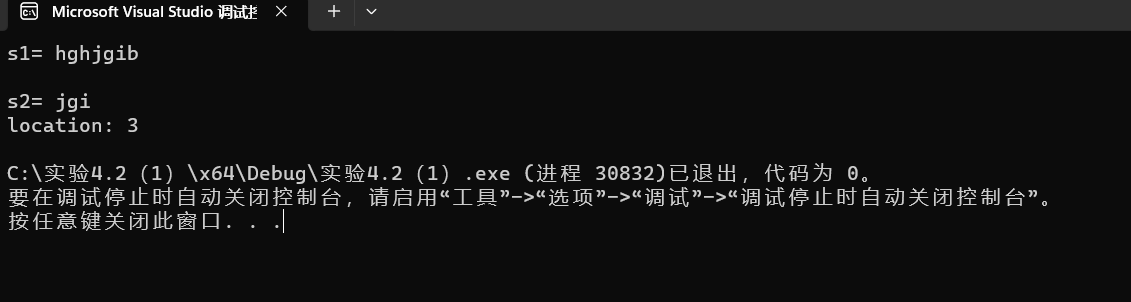
cin >> s2;

int n = indexOf(s1, s2);

cout << "location: " << n << endl;

return 0;

}

****

**4（二）.2.（2）：**

#include<iostream>

#include<cmath>

using namespace std;

int parseHex(const char \*const hexString);

int main()

{

int size=999;

char\* s=new char[size];

cout<<"Please enter a number in hexChar : \n";

cin.getline(s,size);

cout<<"Tne number "<<s<<" in hexChar is "<<parseHex(s)<<" in decimal\n";

delete s;

return 0;

}

int parseHex(const char \*const hexString)

{

int a,sum=0;

a=strlen(hexString);

int\* list=new int[a];

for(int i=0;i<a;i++)

{

if(hexString[i]>='A'&&hexString[i]<='F')

list[i]=(static\_cast<int>(hexString[i])-55)\*(pow(16,a-1-i));

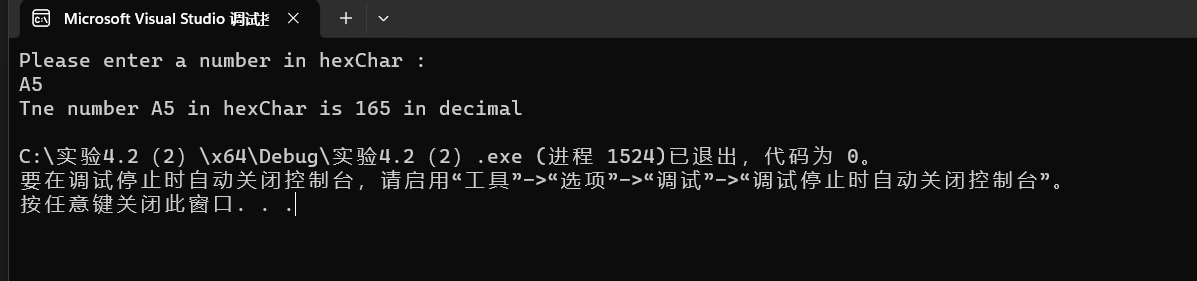
else

list[i]=(hexString[i]-48)\*pow(16,a-i-1);

sum=sum+list[i];

}

return sum;

****}

**4（二）.2.（3）：**

#include<iostream>

using namespace std;

void arrange(int s1[], int size) {

for (int i = 0; i < size; i++)

{

for (int j = 0; j < size-i-1; j++) {

int temp = 0;

if (s1[j] > s1[j + 1]) {

temp = s1[j];

s1[j] = s1[j + 1];

s1[j + 1] = temp;

}

}

}

}

int main() {

int size;

cout << "元素个数为：";

cin >> size;

int\* list = new int[size];

cout << "数组元素为：";

for (int d= 0; d < size; d++) {

cin>> list[d];

}

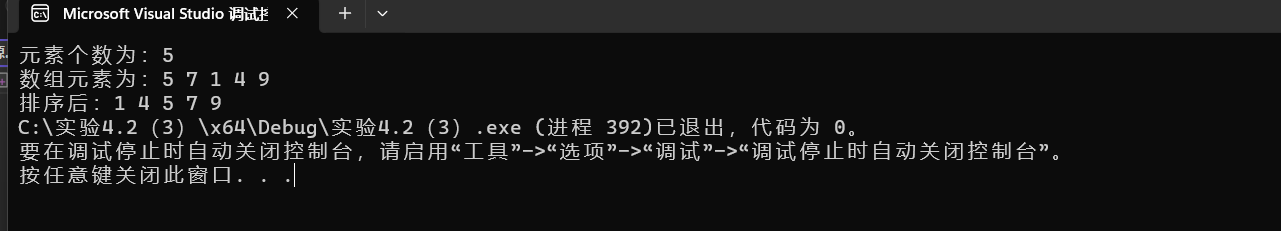
arrange(list, size);

cout << "排序后：";

for (int k = 0; k < size; k++) {

cout << list[k] << ' ';

}

**** delete list;

}

**四、遇到的问题与解决方法**

**问题：（1）开始不知道如何在vs中设置动态数组，导致前面的题并没有使用，把简单的问题复杂化。**

**（2）逻辑不清晰，想不出问题的解决方法。**

**（3）有的题需要利用没有学习的函数，熟练度较低。**

**解决方法：**

1. **学长在微信群里教了如何设置动态数组，自己加以利用。**
2. **想不出来答案不如直接去网上借鉴他人的思路，但一定要学会。**
3. **熟能生巧，多多练习c++的题目。**

**五、体会**

**（1）人外有人，天外有天，不要取得一点小小的成绩就沾沾自喜，比自己努力并且有能力的比比皆是，还需要继续努力。**

**(2)培养独立解决问题的能力，这一点还需要慢慢来，不能操之过急。**